NIST Special Publication 800-XX

Draft Federal S/MIME V3 Client Profile

C. Michael Chernick

National Institute of Standards and Technology
Technology Administration
U.S. Department of Commerce

C O M P U T E R S E C U R I T Y

1901-2001
NIST CENTENNIAL

NIST Special Publication 800-XX

Draft Federal S/MIME V3 Client Profile

Recommendations of the
National Institute of Standards and Technology

C. Michael Chernick

C O M P U T E R S E C U R I T Y

Computer Security Division
Information Technology Laboratory
National Institute of Standards and Technology
Gaithersburg, MD 20899-8930

May 2001

U.S. Department of Commerce
Donald L. Evans, Secretary

Technology Administration
Karen H. Brown, Acting Under Secretary of Commerce for Technology

National Institute of Standards and Technology
Karen H. Brown, Acting Director

Table of Contents

Acknowledgments

NIST would like to thank the many people who assisted with the development of this profile. We are grateful from the support we received from members of the IETF PKIX and S/MIME Working Groups. Special thanks are due to John Pawling, Jim Schaad, Russ Housley, Blake Ramsdell, and Tim Polk.

Draft Federal S/MIME V3 Client Profile

1 Introduction

S/MIME (Secure / Multipurpose Internet Mail Extensions) is a set of specifications for securing electronic mail. S/MIME is based upon the widely used MIME standard [MIME], and describes a protocol for adding cryptographic security services through MIME encapsulation of digitally signed and encrypted objects. The basic security services offered by S/MIME are authentication, non-repudiation of origin, message integrity, and message privacy. Optional security services include: signed receipts; security labels; secure mailing lists; and an extended method of identifying the signer's certificate(s).

S/MIME Version 3 is the latest version of S/MIME. Version 3 is specified in IETF RFCs 2630 thru 2634 ([RFC2630], [RFC2631], [RFC2632], [RFC2633], and [RFC2634]).

The S/MIME specifications were designed to promote interoperable secure electronic mail, such that two compliant implementations would be able to communicate securely with one another. However, implementations may support different optional services and the specifications may unintentionally allow multiple interpretations. As a result, different implementations of S/MIME may not be fully interoperable or provide the desired level of security.

The S/MIME specifications rely on cryptographic mechanisms and public key infrastructures (PKI) to provide security services. If the cryptographic and PKI components that are used to support the S/MIME implementation are sufficiently robust, users can obtain additional assurance that sufficiently strong cryptographic algorithms are used and that procedures are in place to protect sensitive information.

Conformance to this profile helps to assure that S/MIME implementations will be able to interoperate and provide reasonable assurance to users.

The National Institute of Standards and Technology (NIST) - Information Technology Laboratory - Computer Security Division has developed this S/MIME client profile as guidance in the development and procurement of commercial-off-the-shelf (COTS) S/MIME-compliant products. This profile document identifies requirements for a secure and interoperable S/MIME V3 client implementation. NIST is developing tests and testing tools to determine the level of conformance of an S/MIME V3 client implementation with this profile.

This profile does not address requirements for network infrastructure components that implement S/MIME V3, such as mail list agents (MLAs) and secure mail gateways (e.g., security guards). Such systems will have significant overlap but will have additional requirements specific to their function.

The remainder of this document is organized into four principal sections: (a) S/MIME profile requirements; (b) Support for Enhanced Security Services (ESS); (c) Optional Features and Notes on Testing; and (d) References. In addition, there is a non-Normative Annex on Cryptographic Algorithms. The profile requirements describe the minimum functionality required to support secure and interoperable S/MIME client implementations. The support for ESS section discusses requirements for ESS conformance. The optional features and notes on testing section specifies desirable, but optional features for S/MIME client implementations as well as useful notes for testing. These features will be useful to "power" users, but fall outside the core services required to support mainstream S/MIME users. The reference section provides references, while the non-normative annex on cryptographic algorithms provides guidance on cryptographic algorithms.

2 S/MIME Profile Requirements

Important S/MIME interoperability and security features are identified in this section. First, underlying technologies are examined, including cryptography, public key infrastructure (PKI), and formatting of cryptographically protected data. The following subsection describes requirements in these areas. Using these technologies, S/MIME clients perform two fundamental operations: they generate electronic mail messages and process electronic mail messages. The second and third sections describe the minimum functionality for S/MIME message generation and reception, respectively.

Conformant implementations must be able to generate and process signed, encrypted, and signed and encrypted messages as detailed in the paragraphs that follow. Conformant implementations must be able to support all of the mandatory (i.e., MUST of [MUSTSHOULD]) clauses of the IETF RFCs 2630, 2632, and 2633 with the EXCEPTION that implementation of RFC 2631 Diffie-Hellman Key Agreement [RFC2631] cryptographic algorithm mandated in [RFC2630] is NOT required by this Federal S/MIME V3 Profile. However, implementation of the RFC 2631 Ephemeral-Static Diffie-Hellman Key Agreement Method is recommended.

Conformance to this profile also assumes implementations of FIPS approved cryptographic algorithms as specified in Clause 2.1.1.

2.1 Fundamental Technologies

S/MIME relies on three fundamental technologies to format and protect electronic mail messages. These fundamental technologies are cryptographic algorithms, public key infrastructure (PKI), and the cryptographic message syntax (CMS) data format. Correct implementation of these mechanisms is essential to the security and interoperability of every S/MIME client. While these technologies will not be tested in isolation, they will be tested indirectly.

However, not all features available through these technologies are required for S/MIME. To that end, we describe the features that are required by this profile.

2.1.1 Cryptographic Algorithm Suite Conformance

To help ensure that cryptographic functions are correctly implemented, software modules implementing cryptographic functions MUST conform to FIPS 140-1, Security Requirements For Cryptographic Modules [FIPS140-1].

2.1.1.1 Mandatory to implement algorithm suites

To conform to the Federal S/MIME Profile implementations MUST support the following suites of cryptographic algorithms:

ALS1: {RSA (RFC 2313) for digital signature, RSA (RFC 2313) for key transport, SHA-1 hash algorithm, Triple-DES [FIPS46-3] for content encryption} Conforming implementations MUST support a RSA key size of at least 1024 bits. Conforming implementations MUST support RSA algorithms as defined in [RFC2313]. For Triple-DES, when generating messages at least two independent keys MUST be supported using the Cipher Block Chaining Mode (CBC). This algorithm is also known as DES EDE3 CBC.

ALS2: {DSA for digital signature, RSA [RFC2313] for key transport, SHA-1 hash algorithm, Triple-DES for content encryption} Conforming implementations MUST support a DSA key size of 1024 bits.

This profile requires that implementations be able to verify signatures on certificates and messages using either algorithm suite. However it is not required that both algorithm suites MUST be supported for the purpose of generation of digital signatures. Stated slightly differently, this implies that both the RSA digital signature and DSA algorithms MUST be supported for verification, but only one of these algorithms MUST be supported for message generation. It is recommended that implementations SHOULD support both signature algorithms for message generation.

2.1.1.2 Recommended algorithm suites

In addition, conforming implementations SHOULD support the following additional algorithm suites:

ALS3: {RSA [RFC 2313] for digital signature, RSA [RFC 2313] for key transport, SHA-1 hash algorithm, AES for content encryption}

ALS4: {DSA for digital signature, Diffie-Hellman [RFC2631] for key agreement, SHA-1 hash algorithm, Triple-DES for content encryption}

ALS5: {DSA for digital signature, Diffie-Hellman [RFC2631] for key agreement, SHA-1 hash algorithm, AES for content encryption}

Related recommendations on public key sizes:

- If an implementation supports AES, the implementation SHOULD also support RSA public key sizes greater than 1024 bits.
- If an implementation supports AES and D-H, the implementation SHOULD also support D-H public key sizes greater than 1024 bits.

- If an implementation supports RSA or DSA public key sizes greater than 1024 bits, the implementation SHOULD also support SHA-256.

Additional algorithm suites will be identified in the future (e.g., to support SHA-256).

Support of additional algorithm suites, especially using other FIPS approved algorithms (e.g., ECDSA) [FIPS140-1], is encouraged to promote interoperability and backward compatibility. However, the user is cautioned that some older and weaker algorithm suites should be avoided and not trusted.

Further information on cryptographic algorithm suites is provided in Annex A.

2.1.2 PKI Profile Conformance

In order to help ensure that S/MIME V3 implementations might interoperate securely, it is useful to adopt the use of a profile of the X.509 standard. The most widely accepted PKI profile is the IETF Public Key Infrastructure using X.509 (PKIX) profile developed by the PKIX working group. The PKIX profile, Internet X.509 Public Key Infrastructure Certificate and CRL Profile, [RFC2459], (or more recent version) identifies the format and semantics of certificates and CRLs for use on the Internet. Procedures are described for processing and validating certification paths in the Internet environment. The PKIX profile has also been adopted by the Federal PKI Technical Working Group in the "Federal Public Key Infrastructure (PKI) X.509 Certificate and CRL Extensions Profile" [CERTCRL]. Conformance to this S/MIME V3 Profile requires conformance to [CERTCRL].

2.1.3 CMS Content Types

Conforming implementations MUST be able to generate and receive the following CMS Content Types as defined in [RFC2630]:

- Data
- SignedData
- EnvelopedData.

In S/MIME messages the Data content type is only used within SignedData or EnvelopedData content types and always contains a MIME encoded message.

Conforming implementations MUST be able to process nested content types that appear in S/MIME messages. Conforming implementations MUST be able to process an envelopedData content type that contains a signedData content type. Also, conforming implementations MUST be able to process a signedData content type that contains an envelopedData content type that contains a signedData content type. Conforming implementations are not required to process arbitrarily nested CMS content types.

2.1.4 MIME encoding

S/MIME is used to secure MIME content. Traditionally S/MIME implementations have used Base64 MIME encoding for all information. (Base64 encoding is used to transfer binary information through Internet (SMTP) mail because Internet mail was originally designed to transfer ASCII information only, and to ensure that binary information is based through the Internet without modification, it is necessary to translate the information into an ASCII

representation such as Base64.) Unfortunately Base64 encoding applied to binary information causes an increase of message size of at least 33%, and thus represents a waste of network bandwidth, and possibly storage at either end.

S/MIME uses binary representations of data, usually "wrapped" in several layers of ASN.1 and other encodings. While it is necessary to have the outermost binary representation Base64 encoded for passage through the Internet, inner binary representations need not have Base64 encoding applied, although most often Base64 is applied even to inner binary representations.

This S/MIME V3 Client Profile requires that implementations MUST be able to receive messages where inner MIME encodings may be binary rather than Base64 encoded. There is no requirement that S/MIME implementations generate messages with inner MIME encodings without Base64, although S/MIME implementations SHOULD be able to generate messages with Base64 encoding applied only to the outermost binary data.

2.1.5 Mail Access Protocols (POP/IMAP)

To send S/MIME messages over the Internet each end user (e.g., email client) must have one or more email accounts on an SMTP-enabled server (i.e., a standard Internet mail server). Normally, each client also needs a "mail access" protocol, (e.g., POP, IMAP) to support transfer of messages to/from the email server and his/her local computer. With respect to this S/MIME V3 Client Profile, the "mail access" protocol is out-of-scope.

2.2 S/MIME Message Generation

S/MIME implementations MUST be able to generate S/MIME messages that are correctly formatted as per IETF S/MIME V3 specifications and include sufficient information for the recipient to verify the signature or decrypt the data. In addition, these messages should provide enough information to the recipient to allow secure responses to be generated.

2.2.1 Sending signed messages

- S/MIME implementations MUST be able to generate SignerInfo including signed attributes.
- S/MIME implementations MUST be able to generate SMIMECapabilities and signingTime attributes.
- S/MIME implementations SHOULD be able to generate signingCertificate attributes.
- S/MIME implementations MUST be able to include user certificates and appropriate CRLs.
- S/MIME implementations MUST be able to generate multipart/signed (i.e., "clear") messages.
- S/MIME implementations MUST be able to request return signed receipt messages (as specified in [RFC 2634]) where the receipt goes to the message originator. (Requesting a return signed receipt where the receipt is directed to a third party is out of scope for this profile, and thus claims to conformance will not be tested.)

2.2.2 Sending encrypted messages

- S/MIME implementations MUST be able to generate symmetric keys, encrypt messages using these keys, and encrypt symmetric keys using PKCS#1 v1.5 RSA keys as defined in [RFC2313].
- S/MIME implementations SHOULD be able to generate Diffie-Hellman keys and derive pairwise symmetric key-encryption keys (KEK) as defined in [RFC2631], and then use the KEK to encrypt the content-encryption key as specified in [RFC2630].
- S/MIME implementations MUST be able to encrypt a message for multiple recipients.
- When the sender supports more than one method for key management, the implementation SHOULD automatically select an appropriate method based on each recipient's key management certificates attributes (usually obtained from previously received messages).
- When the sender supports more than one method for symmetric encryption, the implementation SHOULD automatically select the appropriate encryption method based on each recipient's SMIMECapabilities attributes from previously received messages.
- S/MIME implementations MUST be able to construct a certification path for the receiver's key management certificate, including CRLs, using LDAP. This includes:
  - the ability to build the path.
  - the ability to fetch the recipient certificate using LDAP.
  - the ability to fetch CRLs using LDAP.
  - the ability to fetch issuer certificates using LDAP

- S/MIME implementations MUST validate the certification path where the end certificate is the receiver's key management certificate or reject the certificate.

2.2.3 Sending signed and encrypted messages

- S/MIME implementations MUST be able to satisfy the requirements for sending signed messages as in Clause 2.2.1 above and MUST be able to satisfy the requirements for sending encrypted messages as in Clause 2.2.2 above.
- S/MIME implementations MUST be able to generate signed and then encrypted messages. This includes generating a symmetric key and encrypting the message.
- S/MIME implementations SHOULD be able to use the application/pkcs7-mime type signed then encrypted messages to reduce message size. It is even more preferable to also combine this with binary encoding of MIME content, again to help reduce message size.

2.2.4 Building MIME header

- S/MIME implementations MUST be able to create properly formatted MIME headers as per [RFC2633] for each of the message types above.
- S/MIME implementations MUST be able to create the intermediate MIME wrappers

2.3 S/MIME Message Reception and Processing

S/MIME implementations MUST be able to receive and process S/MIME messages that are correctly formatted. Recipients MUST be able to verify the signature or decrypt the data using the information provided by the sender. Where necessary, the recipient MUST be able to augment sender-provided information with certificates, CRLs, or status information (e.g., LDAP retrievals of CRLs).

2.3.1 Parsing MIME header

- S/MIME implementations MUST be able to process properly formatted MIME headers.

2.3.2 Receiving signed messages

- S/MIME implementations MUST be able to process SignerInfo including signed attributes.
- S/MIME implementations MUST be able to process both multipart/signed (i.e., "clear") and application/pkcs7-signature MIME Type (i.e., "opaque") signed messages.
- S/MIME implementations MUST be able to acquire certificates by extracting certificates from incoming signed messages.
- S/MIME implementations MUST handle unknown attributes gracefully by accepting the message and informing the user.
- S/MIME implementations MUST be able to construct a certification path for the sender's signature certificate, including CRLs, using LDAP. This is further described in Clause 2.4.4. below.
- S/MIME implementations MUST validate the certification path where the end certificate is the sender's signature certificate or reject the certificate.
- S/MIME implementations MUST ensure that either the SubjectAltName.rfc822Name or PKCS#9 emailAddress in the signer's certificate matches the actual email address used in the received message's "From" or "Sender" field (See [RFC2459] and [RFC2632]). If the addresses do not match or the certificate does not contain any email address, the S/MIME implementation MUST display information to the user to allow the user to accept or reject the message.
- S/MIME implementations MUST be able to generate return signed receipt messages as specified in [RFC2634]. Signed receipts SHOULD be sent to the entities named in the receiptsTo field of the receipt request according to the processing described in Section 2 of [RFC2634]. Note: As specified in [RFC2634], this processing may cause modification in determining if return signed receipts should be sent according to the value of the mlExpansionHistory attribute (if any) contained in received messages.

2.3.3 Receiving encrypted messages

- S/MIME implementations MUST be able to process encrypted messages. This includes recovering the symmetric key and using the key to decrypt the message.
- S/MIME implementations MUST be able to process messages for one or more recipients.
- Receiving agents SHOULD allow a transparent selection of the appropriate private key for decryption of an incoming message when the recipient has multiple certificates (each associated with a private key) used for key management.

2.3.4 Receiving signed and encrypted messages

- S/MIME implementations MUST be able to satisfy the requirements for receiving signed messages as in Clause 2.3.2 above and MUST be able to satisfy the requirements for receiving encrypted messages as in Clause 2.3.3 above when processing messages that are both signed and encrypted.

2.3.5 Processing return receipts

- S/MIME implementations MUST be able to process return signed receipts as specified in [RFC2634] for messages that the implementation generated and are still retained on the client system. Receipts for messages generated by a third party are out of scope for this profile, and thus claims to conformance will not be tested.
- S/MIME implementations MUST be able to match receipts to messages automatically.
- When S/MIME implementations process return signed receipt messages the requirements for receiving signed messages as in Clause 2.3.2 above MUST be satisfied.

2.4 Certificate Processing

S/MIME is a PKI-enabled secure application. Specifically, S/MIME V3 implementations MUST obtain and validate X.509 public key certificates to validate the signature on a message or exchange symmetric key material. If critical PKI features are not present, or are weak, the security of S/MIME implementations will be adversely affected. PKI features are described in detail in [RFC2459], "X.509 Certificate and CRL Profile for the Internet PKI".

The PKI features may be divided into four basic categories. These categories are explained in detail below.

2.4.1 X.509 Certificate Processing

- S/MIME implementations MUST be able to process all critical or optionally critical certificate extensions in the Federal Certificate and CRL Profile [CERTCRL]. S/MIME implementations MUST not reject certificates with unrecognized non-critical extensions. S/MIME agents MUST have the capability to support distinct certificates for digital signature and key management security services for both message origination and reception.

2.4.2 X.509 CRL Processing

- S/MIME implementations MUST be able to process all critical or optionally critical CRL extensions in the Federal Certificate and CRL Profile [CERTCRL]. S/MIME implementations MUST not reject CRLs with unrecognized non-critical extensions.

2.4.3 Path Validation

S/MIME implementations MUST be able to validate a certification path according to [RFC2459], Section 6. S/MIME implementations MUST be able to use X.509 CRLs to establish certificate status for path validation. At a minimum, the following aspects of path validation MUST be implemented:

- certificate policies, policy constraints, and policy mapping
- basic constraints

- name constraints for distinguished names and RFC822 names
- DSA parameter inheritance (if DSA is supported by the implementation)
- processing certification paths with multiple signature algorithms
- name chaining
- signature verification
- validity date checking
- revocation checking by processing CRLs
- key usage/extended key usage
- CRL distribution points
- CRL entry extensions: invalidity date, reason code.

2.4.4 Path Building

Although not described in [RFC2459], path building is an important aspect of PKI-enabled applications. S/MIME implementations MUST be able to construct certification paths between an accepted trust point and a sender's or a recipient's certificate(s). Path building algorithms are not specified in any standard or specification. However, a variety of resources are available to assist in path building using heuristic methods. These resources include standard directory attributes, and a variety of supplementary information in certificate extensions. S/MIME implementations MUST be able to take advantage of this information to provide complete path building services. At a minimum, the following features MUST be supported:

- Use of directory systems.
- CRL distribution points extension.
- Cross certificate pair attribute as specified in [RFC2587].
- CA certificate attribute as specified in [RFC2587].
- Implementations may locate certificates through any of the following methods: AIA extension, SIA extension, or retrieval from a well-known directory. Implementations may locate CRLs through either of the following methods: CRL distribution point extension, or retrieval from a well-known directory.

In addition, either one or both of the following features MUST be supported:

- Use of PKIX Authority Information Access (AIA) extension.
- Use of PKIX Subject Information Access (SIA) extension.

3 Support for Enhanced Security Services for S/MIME (RFC 2634)

The IETF has defined a set of Enhanced Security Services (ESS) for S/MIME in [RFC2634]. The services defined include signed receipts, security labels, secure mailing lists, and an extended method of identifying the signer’s certificate(s). This Federal S/MIME V3 Client Profile does not require conformance to [RFC2634] with the exception that signed receipt requesting, processing and generation is required as specified in Clause 3.1 of this profile. However, mail agents that include support for any of these ESS services may optionally claim support for any or all of these services either in origination of messages or receipt of messages or both. Note that some optional services defined in

[RFC2634] are out of scope for this profile as specified below, and thus claims to conformance will not be tested.

3.1 Signed Receipts

Support for signed receipts is one of the four optional security services defined in [RFC2634]. For the purposes of this S/MIME Client Profile, email agents MUST be able to request, generate and process signed receipts as described in [RFC2634].

- S/MIME agents that originate messages MUST be able to generate signed receipt requests for signed messages as defined in [RFC2634]. (See Clause 2.2.1 above.)
- S/MIME agents that receive messages MUST be able to generate signed receipts for signed messages that they receive containing signed receipt requests as defined in [RFC2634]. (See Clause 2.3.2 above.)
- S/MIME agents that receive messages MUST be able to process signed receipts (including signature verification) as defined in [RFC2634]. (See Clause 2.3.5 above.)
- Mail list agent processing is beyond the scope of this profile. However, S/MIME agents MUST be able to process mlExpansionHistory attributes as defined in Section 2 of [RFC2634].

3.2 Security Labels

Support for security labels is one of the four optional security services defined in [RFC2634]. If an S/MIME implementation claims to conform to the security label service defined in Clause 3 of [RFC2634], then the following requirements are imposed on the implementation:

- S/MIME agents that originate messages MUST be able to generate security labels as defined in [RFC2634].
- S/MIME agents MUST be able to display security labels in received messages as defined in [RFC2634]. S/MIME agents MUST be able to examine the security label on a received message and determine whether or not the recipient is allowed to see the contents of the message OR reject the message.
- The generation and processing of Equivalent Security Labels is beyond the scope of this profile, but may be added to a future profile. However, as stated in [RFC2634], all receiving agents SHOULD recognize "equivalentLabels attributes even if they do not process them."

3.3 Secure Mailing Lists

Support for Secure Mailing Lists ("Mail List Management") is one of the four optional security services defined in [RFC2634]. Mail list agent processing is beyond the scope of this profile but may be added to a future profile. However, S/MIME implementations MUST be able to process received signed messages that contain the mlExpansionHistory attribute as described in Clause 3.1 above.

3.4 Signing Certificate Attribute

Support for Signing Certificate Attributes is one of the four optional security services defined in [RFC2634]. If an S/MIME implementation claims to conform to the Signing Certificate

Attribute requirements defined in Clause 5 of [RFC2634], then the following requirements are imposed on the implementation:

- S/MIME agents that originate messages MUST be able to generate messages which contain a signing certificate attribute as defined in [RFC2634].
- S/MIME agents that receive messages MUST be able to properly process messages which contain a signing certificate attribute as defined in [RFC2634].

4 Optional Features and Notes on Testing

Since the functionality offered by these optional features may be achieved through other means, inclusion of these options in S/MIME implementations is recommended, but not required.

4.1 Generate Application/pkcs7-signature MIME Type (Opaque) Signed Messages

Sending agents SHOULD have the capability to generate application/pkcs7-signature MIME Type (i.e., opaque) signed messages.

4.2 Self-Signed Certificate

Sending and receiving agents SHOULD have the capability to support self-signed certificates.

4.3 Sending CRLs

Sending agents SHOULD have the capability to include appropriate CRLs with outgoing messages (i.e., CRLs which are associated with the sender's certificates).

4.4 Selective Trust of Certificates

S/MIME agents SHOULD provide the capability to selectively trust certificates.

4.5 Acquiring Certificates

S/MIME agents SHOULD have the capability to acquire certificates in either of the following ways:

1) Loading certificates (or chains of certificates) from *.p7c and *.p7m files (file extension types) and messages as defined in [RFC2633].
2) Loading certificates from commonly used (e.g., *.cer and *.crt (binary encoded certificates) file extension types.
3) Lookup of certificates (and chains of certificates) from a LDAP repository

4.6 Importing/Exporting PKCS #12 Credentials

S/MIME agents SHOULD have the capability to import and export PKCS #12 objects as defined in [PKCS#12].

4.7 Notes on Testing and Scope

- Some testing requires “human scoring” because a user interface in involved. For example, as stated in Clause 2.3.2:

"S/MIME implementations MUST ensure that either the SubjectAltName.rfc822Name or PKCS#9 emailAddress in the signer's certificate matches the actual email address used in the received message's "From" or "Sender" field (See [RFC2459] and [RFC2632]). If the addresses do not match, the S/MIME implementation MUST display information to the user to allow the user to accept or reject the message.

There is no efficient method to automate verification that the S/MIME implementation actually displays the mismatch information. Thus, human scoring is required to evaluate the implementations conformance to this requirement.

- Security of operating system and email mechanisms are beyond the scope of this profile. For example, system operators must ensure that the operating system itself is secured, and that the email system is secure. This profile addresses only the security aspects of IETF developed S/MIME extensions.

5 References

[AES] *Advanced Encryption Standard*, http://csrc.nist.gov/encryption/aes/, Draft FIPS, 2001.

[ANSIX9.31] ANSI X9.31-1997, *X9.31 Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (rDSA)*, 1997.

[ANSIX9.44] ANSI X9.44-2001, *Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Factoring-Based Cryptography*. Working Draft, January 12, 2001.

[ANSIX9.57] ANSI X9.57-1997, *Public Key Cryptography for the Financial Services Industry: Certificate Management*. 1997.

[ANSIX9.62] ANSI X9.62-1999, *Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)*. 1998.

[ANSIX9.63] ANSIX9.63-2001, *Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport Using Elliptic Curve Cryptography*. Working Draft, May 8, 2001.

[CERTCRL] *Federal Public Key Infrastructure (PKI) X.509 Certificate and CRL Extensions Profile*, http://csrc.nist.gov/pki/twg/y2000/papers/twg-00-18.xls (Note: In Excel Format), 18 April 2000.

[FIPS46-3] FIPS 46-3, *Data Encryption Standard* (DES), Defines and specifies the use of DES and Triple DES, http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf, November 1999.

[FIPS140-1] FIPS 140-1, *Security Requirements For Cryptographic Modules*, http://csrc.nist.gov/cryptval/140-1.htm, January 1994.

[FIPS180-1] FIPS 180-1, *Secure Hash Standard*, http://www.itl.nist.gov/fipspubs/fip180-1.htm, April 1995.

[FIPS186-2] FIPS 186-2, *Digital Signature Standard*, http://csrc.nist.gov/publications/fips/fips186-2/fips186-2.pdf, Feb. 2000.

[MUSTSHOULD] Bradner, S., *Key words for use in RFCs to Indicate Requirement Levels*, BCP 14, RFC 2119, March 1997.

[MIME] RFC 1341, N. Borenstein, and N. Freed, *Mechanisms for Specifying and Describing the Format of Internet Message Bodies*, June 1992.

[PKCS #1] RSA Cryptography Standard, ftp://ftp.rsasecurity.com/pub/pkcs/ascii/pkcs-1.asc, (See [RFC2313].)

[RFC2313] RFC 2313, Kaliski, B., *PKCS #1: RSA Encryption Version 1.5*, March 1998.

[RFC2459] RFC 2459, *Internet X.509 Public Key Infrastructure Certificate and CRL Profile*, R. Housley, W. Ford, W. Polk, D. Solo, January 1999.

[RFC2587] RFC 2587, *Internet X.509 Public Key Infrastructure LDAPv2 Schema*, S. Boeyen, T. Howes, P. Richard, June 1999.

[RFC2630] RFC 2630, *Cryptographic Message Syntax, defines a cryptographic algorithm independent format for signed and encrypted data*, June 1999.

[RFC2631] RFC 2631, *Diffie-Hellman Key Agreement Method*, defines a variant of the Diffie-Hellman cryptographic algorithm as the mandatory key agreement method for S/MIME V3, June 1999.

[RFC2632] RFC 2632, *S/MIME Version 3 Certificate Handling*, describes how an S/MIME V3 client uses public key infrastructure (PKI) to establish that a public key is valid, June 1999.

[RFC2633] RFC 2633, *S/MIME Version 3 Message Specification*, describes the protocol for adding cryptographic signature and encryption services to MIME data, June 1999.

[RFC 2634] RFC 2634, *Enhanced Security Services for S/MIME*, describes four optional security service extensions: signed receipts; security labels; secure mailing lists; and signing certificates, June 1999.

[SHA-256] *Descriptions of SHA-256, SHA-384, and SHA-512*, http://csrc.nist.gov/encryption/shs/sha256-384-512.pdf, October 2000.

Note: All RFCs are available from the IETF at http://www.ietf.org.

A. Annex A Cryptographic Algorithms (Non-Normative)

The availability and implementation of cryptographic algorithms affect both the security and interoperability of S/MIME implementations. If a message is hashed or signed with an algorithm not supported by the recipient, the recipient cannot verify the signature. If the symmetric key (for message encryption) is derived or transferred with a key management algorithm not supported by the recipient, the recipient will not be able to recover the symmetric key and decrypt the message. If a message is encrypted with a symmetric algorithm not supported by the recipient, the recipient will not be able to recover the clear text.

In addition, cryptographic algorithms offer different levels of protection [ORMAN]. Users of S/MIME products need to make their systems resistant to some predetermined level of attack. That level of attack resistance is the strength of the system. The one-way hash algorithm, digital signature algorithm, key management algorithm and symmetric algorithm SHOULD be at least as strong as the system strength requirements.

Selecting and matching appropriate cryptographic algorithms is a complex task. This document identifies algorithms that offer currently viable levels of security, and identifies suites of algorithms that may be used together.

To help ensure that cryptographic functions are correctly implemented, software modules implementing cryptographic functions MUST conform to FIPS 140-1, Security Requirements For Cryptographic Modules [FIPS140-1] as specified in Clause 2.1.1 of this profile.

A.1 One-Way Hash Algorithms

Hash algorithms map arbitrarily long inputs into a fixed size output such that it is very difficult (computationally infeasible) to find two different hash inputs that produce the same output. Such algorithms are an essential part of the process of producing fixed size digital signatures that can both authenticate the signer and provide for data integrity checking (detection of input modification after signature) in a S/MIME message. For S/MIME V3 the following hash algorithms are identified:

SHA-1: S/MIME agents MUST support the Secure Hash Algorithm (SHA-1) as specified in [FIPS180-1] in agreement with Clause 2.1.1.1 of this profile.

SHA-256: This enhanced secure hash algorithm has been announced. When it becomes widely available S/MIME agents SHOULD be capable of using [SHA-256].

A.2 Symmetric Encryption Algorithms

Symmetric encryption algorithms use the same secret key for data encryption and decryption. The Data Encryption Standard (DES) [FIPS46-3] algorithm using 56 bit keys has been available since the late 1970's and is now considered weak. The much stronger

Advanced Encryption Standard (AES) algorithm allowing up to 256 bit keys has recently been selected and is expected to be widely available soon. In the interim, the triple DES algorithm, which uses multiple passes of the DES algorithm (significantly stronger than DES) is specified for use within S/MIME V3. Thus, for S/MIME V3 the following symmetric encryption algorithms are identified:

3DES: The Triple DES algorithm [FIPS46-3] for encryption and decryption MUST be supported by sending and receiving agents as specified in Clause 2.1.1.1 of this profile. When generating messages at least two independent keys MUST be supported. The Cipher Block Chaining Mode (CBC) MUST be supported. See [FIPS46-3]. This algorithm is also known as DES EDE3 CBC.

AES: The Cipher Block Chaining (CBC) mode of the Advanced Encryption Standard [AES] with 128 bit keys SHOULD be supported by sending and receiving agents when it becomes widely available. (See Clause 2.1.1.2.)

A.3 Digital Signature Algorithms

Digital signature algorithms are asymmetric (using public-private key pairs) encryption algorithms that are used for digitally signing data. Use of signature algorithms in S/MIME V3 provides authentication, message integrity and non-repudiation of origin. For S/MIME V3 the following digital signature algorithms are identified:

RSA: The RSA Public Key digital signature algorithm identified in [RFC2313] MUST be supported in combination with the SHA-1 hash algorithm as specified in Clause 2.1.1.1 of this profile.

DSA: The DSA algorithm defined in [FIPS186-2] MUST be supported in combination with the SHA-1 hash algorithm as specified in Clause 2.1.1.1 of this profile. The object identifier for the algorithm is id-dsa-with-sha1, which is defined in [ANSIX9.57]

RDSA: Implementations MAY support digital signatures using reversible public key cryptography (rDSA) as defined in [ANSIX9.31]

ECDSA: Implementations MAY support elliptic curve digital signatures as defined in [ANSIX9.62]

Note: Currently no standard methods (e.g., appropriate algorithm identifying OIDs) are defined for using RDSA and ECDSA digital signature algorithms with S/MIME.

A.4 Key Management Algorithms

Key management algorithms are used for either key transport or for key agreement in a secure manner.

The following key management algorithms are identified:

RSA (RFC2313): The RSA key exchange algorithm identified in [RFC2313] MUST be supported as specified in Clause 2.1.1.1 of this profile.

Diffie-Hellman: Implementations SHOULD support the Diffie-Hellman Key Agreement method as defined in [RFC2631] as specified in Clause 2.1.1.2 of this profile.

RSA (X9.44): The RSA Key Exchange Algorithm identified in [ANSIX9.44] MAY be supported.

Elliptic Curve Diffie-Hellman (X9.63): The Elliptic Curve Diffie-Hellman Key Agreement Algorithm identified in X9.63 [ANSIX9.63] MAY be supported.

Note: Currently no standard methods are defined for using RSA (X9.44) and Elliptic Curve Diffie-Hellman (X9.63) key management algorithms with S/MIME.

A.5 Algorithm Suites

Secure Hash Algorithms (Message Digest Algorithms), Symmetric Encryption Algorithms, Digital Signature Algorithms, and Key Management Algorithms are used together in algorithm suites to provide a full set of cryptographic security services for S/MIME. Each algorithm suite contains one of each of the cryptographic algorithm types to provide the cryptographic services. It is important that all of the components of each suite provide comparable protection against cryptographic attack. The following list of algorithm suites is identified for use within S/MIME V3:

SHA-1 hash algorithm, RSA (RFC 2313) for digital signature, RSA (RFC 2313) for key transport, Triple-DES [FIPS46-3] for content encryption. Implementations of S/MIME V3 MUST support this suite of algorithms. (See Clause 2.1.1.1)

SHA-1 hash algorithm, DSA for digital signature, RSA (RFC 2313) for key transport, Triple-DES for content encryption. Implementations of S/MIME V3 MUST support this suite of algorithms. (See Clause 2.1.1.1)

SHA-1 hash algorithm, DSA for digital signature, Diffie-Hellman (RFC2631) for key agreement, Triple-DES for content encryption:
All implementations of S/MIME V3 SHOULD support this suite of algorithms. (See Clause 2.1.1.2)

SHA-1 hash algorithm, rDSA for digital signature, RSA [ANSIX9.44] for key transport, Triple-DES for content encryption: Implementations of S/MIME V3 MAY support this suite of algorithms.

SHA-1 hash algorithm, ECDSA for digital signature, ECDH for key agreement, Triple-DES for content encryption:
Implementations of S/MIME V3 MAY support this suite of algorithms.

Recommendations: (These recommendations also appear in Clause 2.1.1.2.)

- If an implementation supports AES, the implementation SHOULD also support RSA public key sizes greater than 1024 bits.
- If an implementation supports AES and D-H, the implementation SHOULD also support D-H public key sizes greater than 1024 bits.
- If an implementation supports RSA or DSA public key sizes greater than 1024 bits, the implementation SHOULD also support SHA-256.

Additional algorithm suites may be identified in the future as advanced algorithms including AES, SHA-256 and extended versions of DSA emerge.